

# Requirements for the skyward r2alpha and r2x ignition system

March 6, 2018

# Contents

<b>1</b>	<b>Overview</b>	<b>2</b>
<b>2</b>	<b>Design goals</b>	<b>3</b>
<b>3</b>	<b>Implementation considerations</b>	<b>4</b>
3.1	Hardware . . . . .	6
3.2	Software . . . . .	9
3.3	Command interface . . . . .	12
3.4	Launch code . . . . .	13
<b>4</b>	<b>Testing</b>	<b>14</b>
<b>5</b>	<b>Upper layers requirements</b>	<b>15</b>

# Chapter 1

## Overview

A safe ignition system is required for the rocksanne r2alpha and r2x rockets. The increased rocket size demands an increase in safety with respect to previous skyward ignition systems.

This document describes the current specifications for the skyward r2alpha and r2x ignition system, which are based on an internal skyward design integrated by taking inspiration also from the following documents:

- Esrange safety manual, document REA00-E60, version 7.

# Chapter 2

## Design goals

The design goals for the ignition system are

1. The design must include a safe/arm switch and an abort device that when not armed or aborted reduce the probability of ignition to an insignificant level.
2. When the ignition system is armed, a minimum of two independent faults are required to cause ignition. A failure in any single component, including the software, must not cause ignition.
3. When the system is armed, only a verified ignition command should cause ignition.
4. The ignition circuit (battery, igniter, and electronics) must be electrically isolated from the command interface (and the rest of the rocket electronics) with a resistance greater than 10Kohms.
5. The igniter terminals must be shorted together until ignition. This requirement applies both in the safe, arm, and abort states.
6. The wires going to the igniter must be twisted and shielded. The ignition system PCBs and assemblies should be shielded as well.

# Chapter 3

## Implementation considerations

Requirement 1 is met by means of:

- a safe/arm switch used to arm the system, that when not armed disconnects the igniter and shorts its terminals
- an electrical fuse that can be remotely blown by means of an abort command. This fuse disconnects the ignition battery.

If the switch is open, or the fuse is blown, there is no electrical path between the battery and igniter, in addition to the two open contacts provided by requirement 2.

Requirement 2 is met through redundancy in the hardware design, and an independent software design. In detail:

- Every component that, upon failure, could cause ignition is duplicated. This includes the microcontroller.
- When the system is armed, two open switches of different nature (e.g: a mosfet and a relay) exist between the battery and igniter. The different nature requirement has been introduced to guarantee that no dependent fault can occur (e.g: a relay may fail mechanically, while a mosfet due to ESD).
- The use of two different microcontrollers, with two different architectures, with software written by two separate engineers and compiled with two different compilers prevents a software bug in the firmware, a bug in one compiler, or a hardware bug in one microcontroller architecture from causing ignition.

Requirement 3 is met through the use of a launch code that have to be manually input into the mission control computer <sup>1</sup>, and transmitted to the rocket to perform ignition.

Requirement 4 is met through the use of optoisolators and/or relays and keeping the command interface and ignition grounds separate.

Requirement 5 is met by wiring the safe/arm switch so as to short the igniter terminals when in the safe state, as well as to disconnect it from the ignition system. In addition, when the switch is in the arm state, the ignition system provides a short circuit as well until ignition.

Requirement 6 is met through the use of appropriate cables and metal boxes for the ignition system and safe/arm switch to ensure shielding.

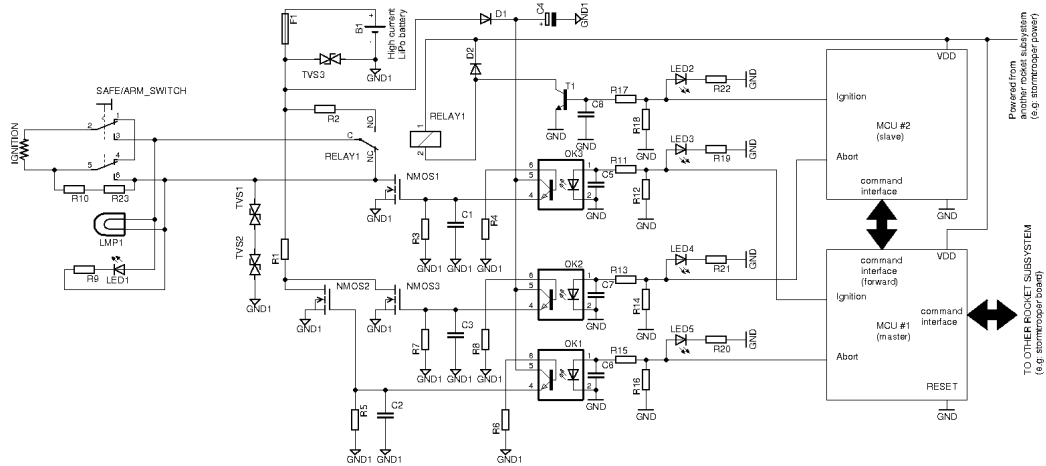
DRAFT

---

<sup>1</sup>The mission control computer is the laptop PC that runs the GUI software displaying realtime telemetry data and also used to send the ignition command

### 3.1 Hardware

A partial schematic showing only the components related to the requirements is shown.



SAFE/ARM\_SWITCH: used to arm the rocket once it is ready on its platform. When in the safe position both terminals of the igniter are isolated from the rest of the ignition system, and are shorted together.

R10 and R23: High value resistors to prevent ESD buildup between the igniter and the rest of the circuit, redundated for safety.

LED1 and LMP1: Should be mounted in the same assembly as the SAFE/ARM\_SWITCH, provide visual indication of the lack of voltage to the operator who is in charge of arming the ignition system. The use of a LED and a filament lamp, two devices built with different technologies has been done to have independent failure modes.

TVS1 and TVS2: Transient voltage suppressors to prevent ESD damage to the ignition system, in particular to NMOS1. Redundated for safety.

TVS3: Transient voltage suppressors to prevent ESD damage when the battery is being connected.

F1: safety fuse, should not blow at the current required for ignition. Blown on purpose when an abort procedure is performed.

R1: fuse blow resistance, should limit the current at a level high enough to blow the fuse, but within the safe level of the LiPo battery. Remember to take into account the  $R_{ds_{on}}$  of NMOS2 and NMOS3. When at least one NMOS is closed the fuse should blow, when both are closed the current should still be safe for the battery. Can be 0 ohm if the short circuit current is safe for the battery.

R2: ignition current limiting resistor, R2 imposes the current to the igniter to a level that causes ignition. Remember to take into account the  $R_{ds_{on}}$  of NMOS1. Can be 0 ohm if the igniter voltage is directly compatible with the battery voltage.

NMOS1 and RELAY1: Ignition switches, when both are closed perform ignition. The use of two components built with different technologies provides independent failure modes for the two switches. In addition, RELAY1 performs the purpose of shorting together the igniter terminals when not performing ignition.

NMOS2 and NMOS3: Abort mosfets, when at least one is closed, the fuse blows. Redundated to allow abort even in case of one failure.

In ogni caso, non sarebbe bene differenziare anche questi? Tipo un relay e un mosfet?

R3,R5,R7: NMOS pulldown resistors. These resistors prevent the gate of the mosfets from being floating when not signaled to close. Together with C1,C2,C3, they also implement an RC filter for glitch suppression.

C1,C2,C3: NMOS glitch suppression capacitors. These RC filters prevent glitches on the driving lines from causing spurious ignitions or abort. Glitches on driving pins are not expected, but these components are added as an extra safety measure.

R4,R6,R8: Optocoupler pulldown resistor. These resistors prevent the base of the optocouplers from floating.

D1,C4: These components provide a power reserve to ensure the gates of the mosfets are provided with an appropriate voltage during actuation. Given the high currents involved during ignition and especially when blowing the abort fuse, a significant voltage drop is expected. C4 should thus be designed to ensure a sufficient gate voltage for the length of time required to blow the abort fuse/perform ignition.

R11,R13,R15,R17, C5,C6,C7,C8: glitch suppression. These RC filters prevent glitches on the driving lines from causing spurious ignitions or abort. Glitches on driving pins are not expected, but these components are added as an extra safety measure. The resistors also serve as current limiting for the optocoupler LEDs and the base of T1.

R12,R14,R16,R18: pulldown resistors. When the power supply to the microcontrollers is not stable, for example at power up or shutdown, the microcontrollers have their pins in high impedance, these resistors provide a stable and safe voltage level.

LED2,LED3,LED4,LED5: Status LEDs, used during the testing and qualification phase to test the proper operation of the firmware of the microcontrollers. These LEDs are expected to be placed directly on the ignition system PCB and are not visible during normal operation.

MCU#1,MCU#2: Two different microcontrollers running the firmware



described below. MCU#1 is the master microcontroller that is connected to the rest of the rocket systems. It therefore forwards the commands received from the external communication interface to MCU#2. For commands which require a reply, such as the status command, it collects the status command response from MCU#2, merging it with its own status prior to transmitting it to the external communication interface.

Both microcontrollers should have a power on reset and brown out reset capability, and it must be enabled. If one of the selected microcontrollers does not have such a capability, an external reset circuit should be provided.

Note: RELAY1 is the only device that shorts together the igniter once the system is armed. It was chosen not to redundate the shorting of the igniter because there are other devices, namely the SAFE/ARM\_SWITCH and the wiring itself that are single points of failure and could present an open circuit to the igniter in case of failure. It should also be considered that the wiring connection to the igniter is shielded, thus providing noise immunity even in the case of a lack of short circuit. It is suggested that RELAY1 be controller by the slave microcontroller, since it has a slightly simpler software (not having to relay commands) and thus it can be expected to have a lower fault probability.

**TODO: evaluate the possibility to simplify the system by removing the redundancy for the abort operation. This means removing NMOS3, OK2 and associated components. The slave MCU would only perform the redundated ignition function.**

Quali sono le motivazioni di questa possibilità? Un abort ridondato imho sarebbe meglio

Non è previsto per il momento un sistema per evitare cali improvvisi di tensione dall'alimentazione ad alta potenza, che sono prevedibili nel momento in cui il circuito scatta e comincia a tirare giù tutti i suoi ampere. Vogliamo mettere un condensatore (magari un supercap), che fa anche da backup per un tempo limitato?

Una volta che il circuito scatta al momento non abbiamo modo per capire se l'igniter sta tirando giù correttamente i suoi 10A e passa o se invece per un qualche motivo (es. fault meccanico sulle piste) non c'è corrente o non ce n'è a sufficienza. Vogliamo pensare a un modo per rilevare questa cosa ed eventualmente segnalarlo ai MCU, così da emettere un abort e notificare?

Abort in questo design è un'opzione attiva: gli NMOS devono essere forzati a condurre, mentre in assenza di controllo sono aperti. Ci piace così o preferiamo che, in assenza di controllo, gli abort siano chiusi?

## 3.2 Software

Two microcontrollers with different architectures should be used (i.e: an AVR and an ARM). The two firmwares should be written by two different engineers, and compiled with two different compilers (i.e: the AVR compiler and the ARM compiler), to prevent unsafe operation also in the event of an error in the compiler.

The master microcontroller should forward the commands received from the external communication interface to the slave microcontroller. For commands which require a reply, such as the status command, the master microcontroller should collect the status command response from the slave microcontroller, merging it with its own status prior to transmitting it to the external communication interface.

Both microcontrollers should have a power on reset and brown out reset capability, and it must be enabled.

Both microcontroller must enable the hardware watchdog feature.

**Note: Both microcontrollers should have the launch code hard-coded and perform the check independently.** An implementation where only one microcontroller checks the validity of the launch command is not acceptable.

The software implemented in both microcontrollers should follow the state machine in Figure 3.2.

S0: Initial state after boot.

The microcontroller must first perform a selftest consisting at a minimum of a checksum of the flash memory content. In case of failure, it should switch to S8

Init GPIO; Ignition\_pin=0; Abort\_pin=0; Init command interface.

S1: Command wait.

Wait until command received, decode command.

S2: Status command received.

Report status.

S3: Launch code received.

Check launch code, if correct perform ignition, else abort.

S4: Perform ignition.

Ignition\_pin=1;

sleep(Trc+Tignition);

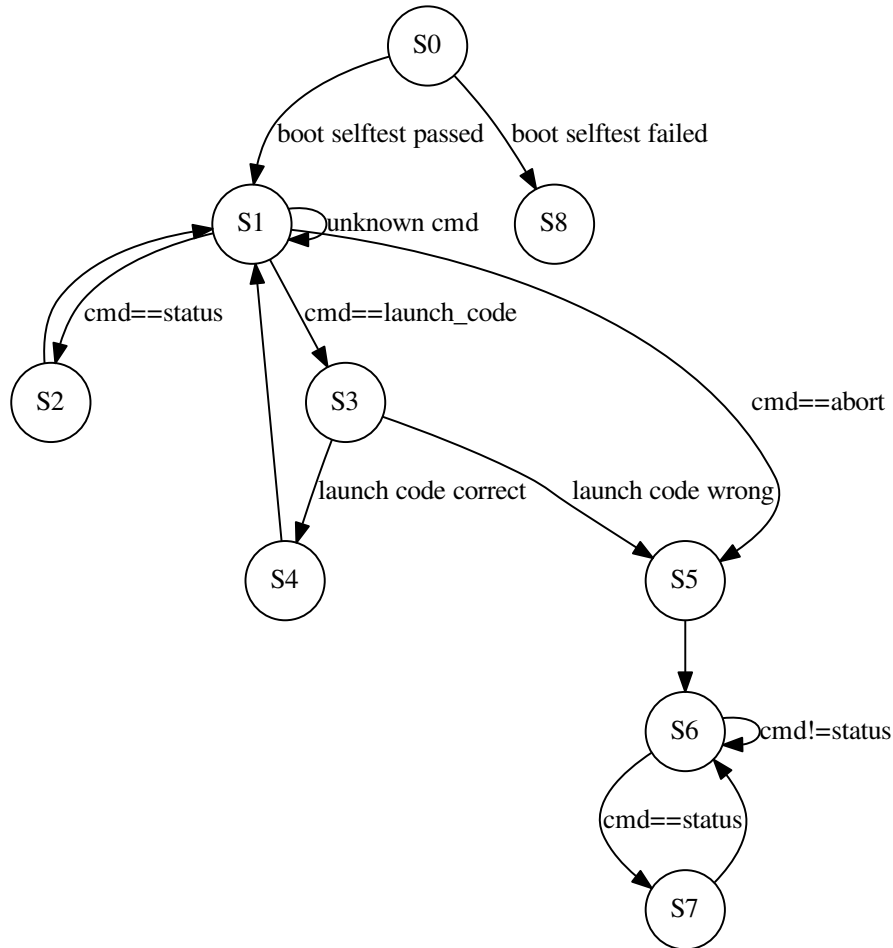
Ignition\_pin=0;

S5: Abort procedure.

Abort\_pin=1;

sleep(Trc+Tfuse);

Abort\_pin=0;



S6: Command wait.

Wait until command received, decode command.

S7: Status command received.

Report status.

S8: Selftest failed.

If the self test failed, the microcontroller is considered incapable of safely performing any action, including the abort procedure. It must therefore be turned off or put in a state that minimizes the probability of performing any unwanted action, such as being locked in an infinite loop with interrupts disabled.

As the state machine shows, the fuse is blown causing an abort procedure when one of two condition occurs:

- an explicit abort commad is received
- the launch code is wrong

The launch code check prevents noise on the command interface or a software bug in other subsystems to cause random data to be interpreted as an ignition command.

This design allows to retry sending the ignition command in the event of a hangfire.

DRAFT

### 3.3 Command interface

The command interface can be either a CAN bus, serial port, or slave SPI, or slave I2C.

Commands, responses and launch codes are expected to be sent as ASCII strings in case of a serial port, or binary data in case of CAN, I2C or SPI.

Appropriate message checks such as CRC or checksums can be added to prevent spurious aborts.

The command interface between the microcontrollers can be either a serial port, or SPI, or I2C. For SPI or I2C, the master microcontroller should be the bus master.

DRAFT

## 3.4 Launch code

The launch code should be a number of at least 16 digits.

An additional checksum digit may be added when inserting the launch code in the GUI of the mission control computer application, which is used to prevent human error when inserting the code, which if unnoticed would cause an abort. The additional check digit should be checked by the GUI and removed in its internal representation of the launch code. The rest of the system should handle the bare launch code, without check digit.

DRAFT

# Chapter 4

## Testing

The firmware of both microcontroller should be tested individually, to make sure that the redundancy does not cause the system to appear functioning even if one subsystem is systematically failing.

The behaviour of the system under noisy power supply (i.e: driving the microcontroller VDD line with a square wave instead of a constant DC voltage) should be checked to assess that no spurious operation occurs.

ESD testing should be preformed.

# Chapter 5

## Upper layers requirements

The launch code should not be hardcoded in the firmware of upper layer microcontrollers, such as stormtrooper, anakin, or the mission control computer application. This prevents a bug in one of those platforms from sending a valid launch code. Hardcoding status and abort commands is instead allowed.

The stormtrooper and anakin should just receive the launch codes from the telemetry link and pass them to the ignition system. The mission control computer application should have a GUI allowing to insert the launch code, which should be input after the rocket is armed **and everyone is at a safe distance from the rocket**.

The operator who is in charge of arming the ignition system must check that the lamp and LED indicators mounted in the same assembly as the SAFE/ARM\_SWITCH are NOT lit prior to arming. **Indicator lights being lit prior to arming indicate a catastrophic failure of the ignition system and must be considered as a NO GO**. This should be written in the launch procedure documents.

The launch procedure document should also include that the battery used to provide power to the igniter must be checked to be fully charged prior to connecting it to the ignition system. A partially charged battery presents a danger because it may not provide enough power to the igniter, causing a hangfire, and/or may not blow the fuse in case of abort.